Ifeanyichukwu Osuji
Summer Internship Report

Categorization of Galactic Images Using a Distributed Architecture

Abstract

Efficient processing of vast amounts of image data has become critical in many areas of astronomy. GalaxyZoo is an online project that uses input from the general public to classify images of over 60 million galaxies. However, as it does not make use of computer assistance in classification, image identification and categorization is exceedingly slow. Although computer vision can be used to automate image identification, it is not always accurate and able to classify images appropriately. Processing large data sets require the ability to store and recall information quickly. In short, rapid identification of astronomical images is hindered by the immense sizes of the image data sets and the inadequacy of current computer vision techniques.

In this project, we designed and developed a software system to address both of these issues. First, our system permits rapid image identification and categorization using computer vision whenever possible, and supplements it with a user interface that allows users both to classify images that were not categorized by the automated process and to gauge computer error. Second, to handle vast amounts of image data efficiently, our system is built using the Apache Hadoop distributed file system architecture on commodity computing hardware. In particular, we use the map-reduce framework that supports efficient distributed processing of large data sets on clusters of computers. We are currently analyzing the effectiveness of our approach and system.

Introduction

As science advances, data sets grow exponentially. The complexity of data processing is also increasing (Managing), sometimes to a degree that it is easier to leave out the computer component, one such application is GalaxyZoo (GalaxyZoo). GalaxyZoo's goal is to classify images of galaxies collected by the Sloan Digital Sky Survey (SDSS). Instead of using a computer system to assist in the classification, they outsource the work to the general public. Even though this is easier than creating a system that will use computer vision, it is not as efficient since it needs to wait for the interaction instead of processing right away. By moving as much processing away from the user as possible, speed will increase.

Large data sets cause two challenges: storage and processing (Managing Scientific Data). In many cases, there is so much data that it is impossible to store on a single machine so more than one is commonly used even if the limit is not reached. By using more machines, not only is more storage added, but more processing power as well (Parallel Database Systems). Usually in the case of data processing, a bottleneck exists on the I/O side (Versatile Storage System) which is not the case of GalaxyZoo where humans are the bottleneck. It is for this reason that it will be more efficient to have the system do as much work as possible. Since data is stored across multiple machines, it will be faster to process all data locally instead of a central location across the whole cluster, this is where Map-Reduce is used (MapReduce: Simplified Data Processing).

As science expands, so does the amount of data that needs to be processed and stored. This is true for both observational and simulation data (Managing). Experiments can produce

petabytes of data which is too much to store on a single machine. Since data is stored on separate machines and possible different physical locations, the possibility of total data loss is decreased. Usually replication of data is also performed so that it is protected even more from natural disaster or machine failure.

Even though the speed of computer processors are growing, the amount of processing time needed is growing even faster (Managing). By increasing the number of processors used and processing data in parallel, the total run time will be decreased. This greatly increases the type of processing available because even extremely large data sets become manageable.

Infrastructure

Figure : Processing architecture with processing flow

Apache Hadoop Distributed File System (HDFS) is an open source project designed to allow for the easy storage of large data sets using commodity hardware. The main goals of HDFS include, being operable on commodity hardware, being able to efficiently store and retrieve large data sets, and being fault tolerant from hardware failure. HDFS is not a kernel level file system such as FAT32, but is a java program run in user space. (Dorthakur, 2009) HDFS is designed to run using multiple machines, Facebook, for example; is uses HDFS on thousands of machines to store their ever increasing data storage needs.

HDFS is designed for use with extremely large files, gigabytes to terabytes of size. Optimized for high bandwidth rather than low latency, HDFS is better suited for batch processing rather than interactive usage. (Dorthakur, 2009) It is designed to work with the Map/Reduce framework. This allows for the easy parallelization of data processing of the large datasets stored in HDFS. By using the map/reduce framework, we are able to process the large number of images in a parallel fashion, thus reducing the overall time it takes to categorize large sets of galactic images. (Dean & Ghemawat, 2004)

We are using an eight-node HDFS cluster. It is built with seven 3.2GHz computers and one 2.8GHZ computers with Pentium D processors and 1GB of ram and 80GB hard drives. The cluster is built on the Fedora 13 Linux operating system and is networked via a 100Mbps Ethernet switch. A DHCP and DNS server on a separate machine is running Windows Server 2008. The NameNode computer in the cluster is also running an Apache http server, a MySQL server and a php server. The total storage capacity of our cluster is approximately 570GB.

The HDFS cluster is filled with over 10,000 of galaxy images. Using Map/Reduce, we can use our computer vision algorithm to process the images in parallel. The resulting categorization is stored, along with metadata, inside a MySQL database. The web server contained the user interface that will access the MySQL database for categorization information and can retrieve the images it wants directly from the cluster. The HDFS cluster runs a basic web server, which allows for the basic monitoring of the cluster, view live and dead DataNodes, status of running jobs, available cluster space and allows for the files in the cluster to be browsed and downloaded.

To run the Map/Reduce program, several java classes were constructed. We created three sets of input and output streams to allow for the reading and writing of images from the cluster, a local system, or a remote ftp or http server. By default, the Hadoop Map/Reduce framework is designed to process text-based files and split these files up per line before running them through the Map/Reduce, this would not work for our purposes. Therefore, we had to change the default

Map/Reduce configuration to use our own code in order to prepare the images for processing without splitting each image into smaller components. Once this is achieved, the framework prepares a mapper job for each individual image, which then runs through our computer vision code. Either the computer vision returns the category of the image as an elliptical or a spiral galaxy, or undefined if it was unable to identify that image. The results are then sent to our MySQL database for usage by the user application.

The distributed system is only used to store images; the metadata is stored in a MySQL database residing on the web server. The web interface will use Hadoop's HFTP which allows read-only access to files through HTTP. An Apache web server was also implemented to handle the web interface that will be used to make corrections to data and augment the categorization. Since the data is going to be changing, PHP was used to create the dynamic websites. PHP also has the ability to query the MySQL database to recall and alter metadata. To keep all data as central as possible, the web server resides on the NameNode.

Image Processing

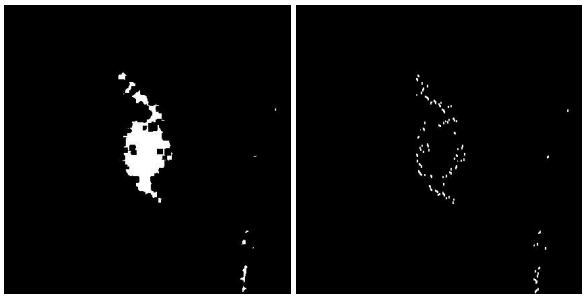

Figure : Stages of computer vision

One part of the system is the image categorization program. This program's job is to use computer vision techniques to try and pick out whether the galaxy in the center of the given image is a spiral galaxy or an elliptical one. A spiral galaxy is one that has a spiral arm pattern, while an elliptical galaxy appears to be more or less a uniformly distributed cluster of stars, though it may contain a few denser clusters. (Aprajita and Phil, 2010)

For our purposes we defined three different possibilities for a galactic image: spiral, elliptical, and undefined. In order to do the final check, we first have to process the images. The first step is to perform a morphological close operation, an image erosion followed by an image dilation, in order to reduce the noise in the image. In the next step we convert the image to a binary image, and image where every pixel has either a 1 or a 0 value, to further reduce noise and also reduce the complexity of the image. After this, we can begin to take a harder look at the contents of the image. By applying a Sobel operator, we can pick out a rough outline of the objects in the image, and it is this image that goes into the final Hough Transform.

The Hough transform can be used to find out where the edges and boundaries of an object in an image are and put them in a form that a computer can work with. In this case we apply an elliptical Hough transform in order to find the largest circle. We then look at the confidence that the transform has in the circle. The confidence is defined as the value at the Hough transform maximum, divided by the maximum that the transform could have at any point; in this case the radius of the circle it found. (Abramoff, 2004) This means that the larger the confidence, the more stretched out the object in the image would be. We tested the confidence of the image against a threshold of one hundred. If it were larger than that we said that it was elliptical. If it were smaller than that we said that it was a spiral galaxy. If the object identified has a confidence of zero, or the object is not close enough to the center of the image, then it is undefined. An undefined image presents a special case where the image is reprocessed without the Sobel operator because this sometimes allows the Hough transform to have a better look. If after this the image is still undefined, then it is left with that categorization.

The website is set up in a way that is hopefully user friendly. Upon entering there are a few different options: classify those that computer vision has not, verify those that were already classified, or view the different categories (which also has a reclassify option).

To classify those that computer vision has not yet, the website will query the database for any entries that do not have a classification. They will then be displayed to the user and they will choose images that belong to one category and also select what category they belong to. To verify, the user will select what category they'd like to observe, then similar to the previous process, they will classify the images they are shown. Users also have the option to view the images from specific categories. They will be broken up to so many on a page so that load time is kept to a minimal for each page.

Figure : Example of the web interface that is used to help classify images

At this point, the user is needed to augment the categorization process. Unfortunately, even the small part of GalaxyZoo's work we're focusing on, needs some human interaction because the computer is not as accurate as human eyes. By allowing human interaction to fix errors, process time increases due to their work speed but will create a more accurate result set when all processing has been completed.

Performance Evaluation/Results

The system processed 10,353 images with the following results: 1,147 spiral, 1,754 elliptical, 7,444 could not be decided, and 8 did not receive any categorization for unknown reasons. There was a 28% classification rate. This is a little lower than was expected but it limits the amount of errors it claims as true, letting what it can't make a decision on to be classified by the user. After the computer vision algorithm did its part, the web application was tested. The rest of the data was not processed with it but it was proven that it does work.

Related Work

Sloan Digital Sky Survey (SDSS) is a long term project which is being run at Ohio State University and Johns Hopkins University. The purpose of this project is to create a 3D map of the night sky. The telescope, located at Apache Point Observatory, NM, scans the sky to collect raw data which is transferred to another location to be processed. In this raw data are stars,

quasars, and galaxies (Managing Scientific Data, ACM June 2010 vol 53 no 6). Creating the 3D map does not require the knowledge of what all the objects are but this information is important to other groups of scientists.

GalaxyZoo was created to do the needed identification of galactic bodies, mainly the different shapes of galaxies. Instead of creating a computer based identification system to identify the galaxies, they outsourced the work to the public via a web interface. Upon signing up to help, users are given a short tutorial on what makes galaxies different from each other. After this tutorial, they are shown an image of a galaxy which they answer a few simple questions on by clicking the appropriate button (GalaxyZoo). This gives researchers valuable information that is used to answer questions about the universe we live in.

Many applications that require storage and processing of large data sets use some form of distributed system architecture. Facebook uses HDFS for storage and uses Hive, a module that uses Hadoop to give it database functionality. Since the database queries take place as map-reduce processes and the amount of data is extremely large, no queries are used for online processing. Instead, they incorporate caching algorithms to serve data quickly to users (High Performance at Massive Scale – Lessons learned at Facebook).

Google uses another distributed architecture called Big Table. They have a master server which requests are processed through then instead of tunneling data from the storage servers through the master, the master sends back to the client where the data is stored and it is retrieved directly (BigTable/Google File System).

Conclusion and Future Work

It was not determined how many of the images were correctly categorized. With a better computer vision algorithm the number of images that receive a category will increase and the reliability of those categories will also be higher. This trial determined that it is possible to use a distributed file system, parallel processing, and computer vision to decrease the amount of human interaction required in a categorization system.

Processing with computer vision, it is also possible to create a semantic language. This uses two binary matrices: log-image and image-feature (Data Mining article). Computer vision will create the image-feature matrix which will pull out features of the images and store them so for each image, it will form a relation between the images and what the images are. The user will then construct the log-image matrix by selecting images that fit a category that the user defines.

When the matrices are combined, a log-feature image is created. This can be used to classify the rest of the images that weren't classified in the logs already. It will search the rest of the images for features that correspond with a log.

A more accurate computer vision algorithm can also be made so less human interaction is needed. By using less human interaction, the overall process will become more efficient.

Also more categories can be implemented so a more precise classification can take place. This is where the binary matrices will be useful since there are multiple options with more unique features.

Works Cited

Anastasia Ailamaki, V. K. (2010). Managing Scientific Data. *Communications of the ACM* , 68-78.

Aprajita and Phil. (2010). *How to Take part*. Retrieved from http://www.galaxyzoo.org/how_to_take_part

Abramoff, M.A. (2004). *:class houghtransform*. Retrieved from http://bij.isi.uu.nl/ImageJ/api/registration/HoughTransform.html

Dean, J., & Ghemawat, S. (2004, December). *MapReduce: Simplified Data Processing on Large Clusters* . Retrieved June 2010, from Google Research Publications.

Dorthakur, D. (2009, 04 09). *HDFS Architecture.* Retrieved 06 2010, from Apache Hadoop: http://hadoop.apache.org/common/docs/r0.20.0/hdfs_design.pdf